An Efficient High Quality Color Transformation

Ingeborg Tastl[1], John Recker[1], Yao Zhang[1,2], Giordano Beretta[1]
[1] Hewlett-Packard Laboratories; Palo Alto, CA / USA
[2] Currently at University of California, Davis, CA / USA

Abstract

This paper describes a new method to select a non-uniform set of sample points specifying a color transformation in a more accurate way than achievable through the same number of uniformly distributed sample points. Furthermore, a specific accuracy can be achieved with far fewer non-uniform sample points. This has direct implications on the real-time processing performance and on the resources (ink, media, time) needed if the set of sample points induces a customized test chart for printer characterization. Although many practitioners implement color transformations as a piecewise linear interpolation using an n-dimensional Delaunay tessellation, this paper discusses the potential to achieve better results with a data-dependent tessellation.

1. Introduction

Our company strives to integrate new imaging technologies, high quality products, and superior user experience with an open, de-facto industry standard color management system, providing outstanding color reproduction in a heterogeneous environment. Commercial printing customers expect solutions which first enable them to reproduce colors as accurately as possible on one device, second enable them to provide a consistent color appearance across a wide range of printing devices, and third enable them to produce pleasing reproductions using the full capabilities of each printing system. Today customers are no longer simply interested in individual printing devices, they want flexible and scalable solutions capable to address a number of simultaneous requirements. This paper focuses on how to achieve efficient, high quality color transforms.

The motivation for using a Graphic Processing Unit (GPU) in a RIP is to address the speed demands of digital presses in an effective and scalable form. The abundant computational power can be used to overcome long-standing constraints on color transforms. Traditionally, non-linear color transformations (e.g. from CIELAB to CMYK) are approximated by multidimensional uniform color look-up-tables (CLUTs) applied in real-time during the ripping process to large documents (containing images, graphics and text). These CLUTs are used for speed purposes: Finding for each color pixel in an image an enclosing cube or tetrahedron in a uniform 3D CLUT and using values at the corresponding vertices to perform linear interpolations is very fast. Those CLUTs might be stored with some other simple syntax elements in ICC device profiles. Although by combining 3D CLUTs with 1D LUTs it is possible to encode data on a *non-uniform* grid, the data still has to be on a *regular* grid.

This implies several restrictions: First, the data on the regular grid has to be derived by some method (interpolation or inverting a printer model) from the original measurement data describing the device characteristic. Second, special attention has to be paid to ensure a high-quality reproduction of colors that are inside a device gamut, but close to the border. Third, if a regular CLUT is used to approximate a non-linear color transformation, a common way to increase accuracy is by increasing the number of entries in the CLUT, which can be expensive and is inefficient (uniform sampling increases samples in both over and under-sampled parts of the function). Last, but not least, there is evidence from sampling theory, computational geometry, and publications in color science [1, 2], suggesting it is advantageous to allocate more sample points to areas of high curvature.

Monga et al. [2] proposed an iterative "sort-select-damp approach" where they calculate the significance of all node points based on curvature and an input importance function. At each iteration they select the node with highest significance. To avoid selecting new points too close to previously selected points, they introduce a fixed damping function. The solution proposed here follows a similar iterative approach, but integrates the dampening effect into the error function. Furthermore, the selection of a new point is determined by both local curvature and the tessellation achieved in previous steps.

This paper proposes a simple and efficient node selection algorithm, generating a data-driven test chart (non-uniform sampling of the device space) whose corresponding device independent and dependent color values are used directly to perform the color transformation. Details and results from initial experiments will be reported in Section 2. Starting from a uniform CLUT it is straightforward to construct n-dimensional simplices (cubes or tetrahedra if n=3) to perform an image's linear interpolation for individual color pixels, but for non-uniform sampling nodes a tessellation has to be calculated explicitly. Color scientists commonly use n-dimensional Delaunay tessellations to achieve well-shaped simplices. However, several publications [1, 4, 5] in computational geometry and mesh refinement indicate that long, thin simplices can actually be desirable and result in a smaller approximation error than simplices obtained by a Delaunay tessellation. Section 3 will discuss the results we achieved by applying our error metric defined in Section 2 as a cost function for a tessellation optimization for an analytical 2D function and for a plane within a printer gamut. Section 4 concludes the paper.

2. Point Selection

The problem can be formulated as a mathematical optimization problem. Test patches from a uniformly sampled device space (e.g. pRGB) are printed and measured (e.g.CIELAB). These samples form a superset from which we select a certain

number of "significant" samples, which, at each iteration, reduce the overall mean squared error. Those samples will be used for an efficient color transformation. Furthermore, the samples define a device and media specific test chart available for customers to characterize their specific devices.

Let $f : R^m \rightarrow R^n$ represent the transform from the printer color space R^m to a device independent color space R^n. This function is approximated by a piecewise linear function f_S described by a large set of samples $S = \{p_i : p_i \in R^m, i = 1,2...l\}$. Our goal is to select a subset of S, $S_{sub} = \{p_j : p_j \in R^m, j = 1,2...k, k \le l\}$, so that the error, i.e., the Lp distance between f_s and f_{sub} is minimized. The Lp distance is defined as

$$L_p = \|f_s - f_{sub}\|_p = \left[\int |f_s - f_{sub}|^p \, dv\right]^{1/p}$$

where v is a multidimensional input variable; $\|f_s - f_{sub}\|_1$ is called *integral absolute error (IAE)* and $\|f_s - f_{sub}\|_2$ is called *integral squared error (ISE)*.

In the 1D case, a curve is approximated by line segments and the error is equal to the area between the line segments and the curve. Mathematically, the following error has to be minimized:

$$Error = \int_{x1}^{x2} |f_S(x) - f_{sub}(x)| dx$$

In the 2D case, a surface is approximated by a triangle mesh, and the error is the volume between the triangle patches and the surface. The following error has to be minimized:

$$Error = \int_{y1}^{y2}\int_{x1}^{x2} |f_S(x,y) - f_{sub}(x,y)| dxdy$$

In the 3D case, a 3D volume is approximated by a set of tetrahedra and the error is the mass between the tetrahedra and the volume. The following error has to be minimized:

$$Error = \int_{z1}^{z2}\int_{y1}^{y2}\int_{x1}^{x2} |f_S(x,y,z) - f_{sub}(x,y,z)| dxdydz$$

We now consider the specific implementations of these concepts in 1D, 2D and 3D.

2.1 Point selection algorithm for 1D

The algorithm's goal is to select sample points on a function f, so that the piecewise linear approximation function f_{sub} defined by the selected sample points leads to the smallest possible error. We propose an iterative algorithm adding one point at each iteration. Starting with two sample points, a function f and the piecewise linear approximation using the two sample points, the original error is the area between the two functions, visualized in *Fig. 1a*. Adding a sampling point reduces the error as visualized in *Fig. 1b*. It can be proven that for the depicted example, the point furthest away from line *AB* results in the minimum error. Although the furthest point may not be optimal for an arbitrary function, it remains a good choice. We define an iterative solution to approximate *f(x)* by line segments:

1) Create an initial point set S_{sub}, consisting of the 2 endpoints of f, and a set of line segments L, consisting of 1 line segment whose end points are the two points in S_{sub}.

2) For each line segment $l_i \in L$, select the point $p_i \in S$, which is on the part of function f, which the line segment l_i tries to approximate, and which is the furthest point to the line segment l_i. Calculate the area α_i of the triangle defined by p_i and the two end points of l_i.

3) Find the maximum $\alpha_j = \max\{\alpha_i\}$ and add the corresponding point p_j to the subset S_{sub} and substitute the line segment l_i with the line segments p_jA and p_jB where A and B are the end points of l_i.

$$S_{sub} = S_{sub} \cup \{p_j\}$$
$$L = (L - \{lj\}) \cup \{(p_j, A) \cup (p_j, B)\}$$

4) If the number of elements in S_{sub} is equal to a threshold, terminate the algorithm, else return to 2.

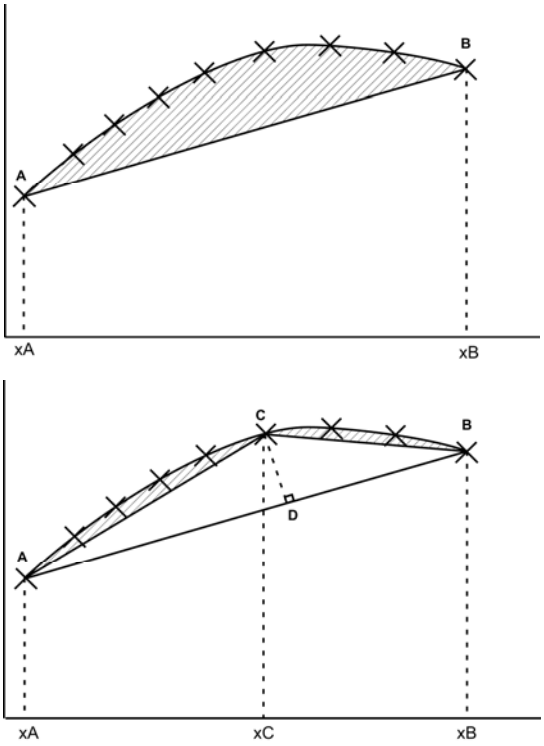

Fig. 1a, b: Approximation error (shaded area) before and after adding one point.

Point p_j corresponds to point (xC,C) in the figure above.

2.2 Point selection algorithm for 2D and 3D

The 2D case is similar, except that we select the point on the function p_i furthest away from a specific triangle. The error metric calculates for each triangle the volume of a tetrahedron whose base is the triangle and whose height is the distance between point p_i and the triangle. The point p_j corresponding to the tetrahedron with the maximum volume is selected and the base triangle is replaced by 3 sub-triangles.

The error metric can be expanded to the 3D case, where for each tetrahedron the point p_i is selected which maximizes the mass for that tetrahedron. The mass is defined as the product of the volume of the tetrahedron and the interpolation error for point p_i. Then, the maximum of the masses for all tetrahedrons is calculated and the corresponding point p_j is selected. The tetrahedron enclosing the selected point is then subdivided into 4 tetrahedra.

Scaling the interpolation error by the distance between neighboring sample points naturally balances the conflicting needs of densely sampling a function in regions of high curvature and distributing samples across the entire function. This contrasts with the dampening function approach as used, for example, by Monga.

Applying the above mentioned algorithm results in a non-uniform sampling of the device space $(x,y,z) \in pRGB$, which takes the curvature of $f(x,y,z) \in CIELAB$ into account. It is a data-dependent node selection process, which aims to select nodes minimizing the approximation error.

2.3 Results of the point selection algorithm

Initial test results show a significant color accuracy improvement for a characterization using a similar number of non-uniform samples (optimized test chart) versus uniform samples (uniform test chart). Furthermore, an optimized test chart generated for a specific printer also outperforms a uniform test chart when used on a different printer unit of the same model.

The experiments were performed using two HP Z3100 large-format printers and a roll of HP Premium Instant-Dry Satin Photo paper. We printed a 13×13×13=2197 uniform RGB test chart on *printer A*, measured the CIELAB values using a spectrophotometer, and used our algorithm to select 3 subsets of 125, 512 and 1000 color patches. Then we printed those 3 optimized test charts and 3 uniform test charts (5×5×5, 8×8×8, 10×10×10) on *printer B* and generated 3 ICC profiles containing a 33×33×33 LUT.

Fig. 2 describes the performed accuracy tests.

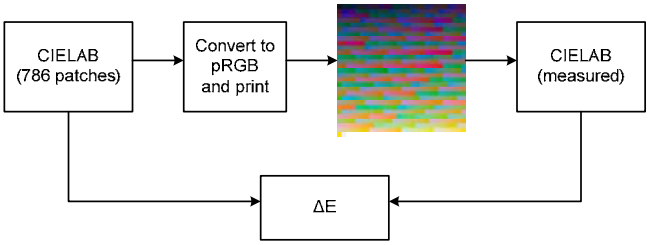

Fig. 2: Testing of the Accuracy of the transformations.

1) We applied the 3 **ICC profiles** to a test data set, printed the patches, measured the CIELAB values, and calculated mean ΔE_{ab} (1976) errors. This data set is referred to as **"uniform, ICC profile"**, indicating a uniform test chart in device color space, from which an ICC profile was built (first interpolation) and then applied to the test data set (second interpolation).

2) We used the **non-uniform chart** data (125, 512 and 1000 patches), generated a Delaunay tessellation, calculated the device dependent values for the test data set through tetrahedral interpolation, printed the patches, measured the CIELAB values, and calculated mean ΔE_{ab} (1976) errors. This data set is referred to as **"non-uniform, direct"**, indicating a non-uniform test chart in device color space, which is directly used to transform the test data set (only one interpolation).

As test data set we used a 19×19×19 uniform grid in CIELAB and selected the CIELAB values (786) inside the printer gamut. We could have selected a different data set — the only important point is to use a test data set different from the set used to describe the device characteristic.

Fig. 3a shows the mean ΔE_{ab} (1976) errors for the test data set and *Fig. 3b* shows the 95 percentile ΔE_{ab} (1976) errors for the test data set. The non-uniform test charts clearly lead to higher accuracies for 125, 512 and 1000 test samples. Furthermore, the data in *Fig. 3* represents a real-world scenario where an optimized test chart was created for one printer and then used and tested for another printer of the same type. The average difference for 1000 patches printed on printers A and B was 1.24 ΔE_{ab} as illustrated in *Fig. 3*. This means that our point selection algorithm is quite robust.

Analyzing the results from a different viewpoint, the data indicates that a predetermined accuracy can be achieved with a smaller sample number if a non-uniform sampling scheme is used versus a uniform one. A smaller amount of samples is particularly important for the direct color transformation method, due to the dependence of the performance on the number of tetrahedra, which is dependent on the number of sample points. *Fig. 4* illustrates the performance for the color transformation of one large document when different sample numbers are used. It exhibits a linear scaling of time with number of samples. For commercial printing solutions both speed and accuracy are important, but can be traded against each other dependent on the needs of a specific type of print job. High quality photo-books require a different accuracy than direct mail.

There are several applications for the described point selection algorithm: On the one hand, a color transformation can be specified in a more efficient (fewer points, faster processing) and accurate way. On the other hand, those non-uniform test charts containing significantly fewer patches, can be provided to customers to build color transforms for their specific devices. In this scenario having to print a test chart with fewer samples means using less resources (ink, media, time), which is always critical.

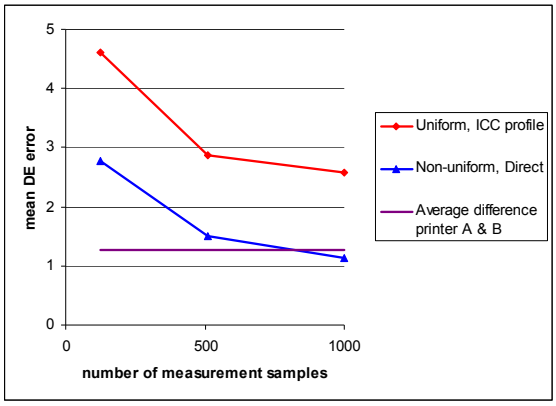

Fig. 3a: Results (mean ΔE) for uniform and non-uniform test charts.

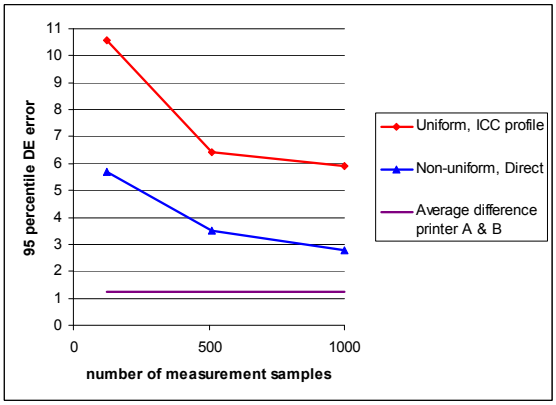

Fig. 3b: Results (95 percentile ΔE) for uniform and non-uniform test charts.

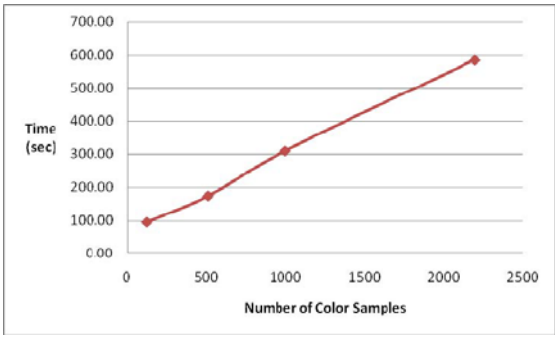

Fig. 4: Processing time for direct inversion.

3. Tessellation optimization

Having a uniform or non-uniform set of samples in the device space, which are printed and measured, will result in a set of non-uniform distributed nodes in the device-independent space (e.g. CIELAB). For printing purposes this set of scattered data points is used to transform an image's colors from the device-independent into the device dependent space. There are many methods to perform a scattered data interpolation, as discussed in detail in [1]. In the case of a tetrahedral interpolation, first a complete tessellation is performed. However, this tessellation is not unique and will lead to different results. Engineers often perform a straightforward tessellation in the device space in the case that a uniform test chart has been used, and simply use the tessellation in the device-independent space. This practice ensures a complete tessellation of the device gamut even when it is concave.

In the case of a non-uniform test chart, color scientists usually apply an *nD* Delaunay algorithm, as it provides well-shaped simplices [3], which are as equiangular as possible. In detail, the algorithm maximizes the minimum angle in a tessellation. Although this tessellation has many desirable features, it also has a couple of drawbacks: First, a Delaunay tessellation is always convex, an issue for printer gamuts because they are often concave in CIELAB. Second, several publications in applied mathematics [4, 5], geometric modeling, and finite elements have shown that a data-dependent tessellation can often significantly improve the quality of the approximation and that despite common belief, long thin triangles can be good for linear interpolation. Data-dependent tessellations not only take into account the location of the points (x,y), but also the function values of the point $f(x,y)$ and some known characteristic or requirement of the function being approximated. Specifically, long thin triangles are well suited to approximate a function $f(x,y)$ having a preferred direction (e.g. large second-directional derivate in one direction, compared to another direction) and the long sides of the triangles are in the direction of small curvature for $f(x,y)$.

Such a data-dependent triangulation can be constructed by starting with an initial tessellation, defining a cost criterion for each edge, and then iteratively evaluating the cost of an edge in a quadrilateral and swapping it when the cost of the alternative edge is lower. We tested this approach for the 2D case, where the edge swapping is driven by the error metric from Section 2, and the Delaunay tessellation is replaced with the optimized tessellation.

Given a set of points $p_j(x,y)$ and the corresponding function values $f(p_j)$ we can apply the following algorithm:

1) Perform an initial Delaunay tessellation of the points p_j, resulting in a list of triangles.

2) For each triangle, test whether the triangle and one of its neighbors form a convex quadrilateral. If that's the case, test whether swapping the internal edge reduces the cost function. If yes, swap the edge and replace the two initial triangles with the two new triangles and put them at the end of the list. See *Fig. 5*.

3) If no further edges can be swapped, or if a certain number of iterations have been achieved, stop; otherwise go back to step 2.

Using the 2D error metric from section 2.2, the cost function selects the pair of triangles generated by edge swapping, that result in the smallest summed error.

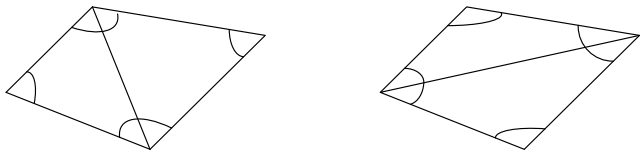

Fig. 5: Edge swapping in a quadrilateral.

For illustration purposes, we selected a test function f(x,y) from [4] with a sharp drop running diagonally across a square (*Fig. 6*),

$$f(x,y)=\frac{\tanh(9y-9x)+1}{9}$$

calculated the functional values for a regular grid of 11×11 and selected 36 points using the algorithm described in Section 2.2, and then performed both a Delaunay and the data-dependent tessellation described above. The results are visualized in *Figs. 8* and *9*. For comparison *Fig. 7* visualizes the results for a Delaunay triangulation of 36 regular points. *Fig. 9* shows that the triangles are thin in the direction of strong curvature and long in the perpendicular direction. That means that our cost measure for edges does give the expected results. The average error for the test function for the test data set of 169 values is *avg.ΔE= 0.0018* for the Delaunay tessellation and *avg.ΔE=0.0007* for the data dependent tessellation.

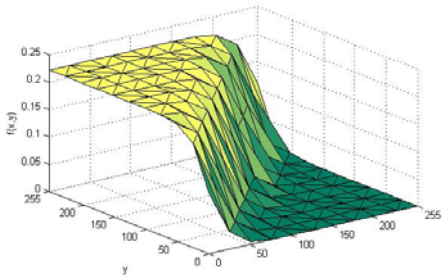

Fig. 6: Function used for test purposes.

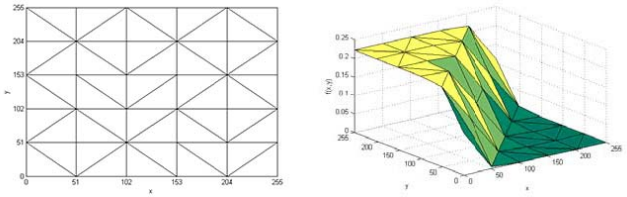

Fig. 7: Delaunay tessellation of 36 uniform distributed points.

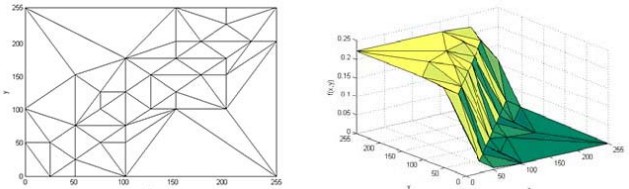

Fig. 8: Delaunay tessellation of 36 selected points, avg.ΔE= 0.0018.

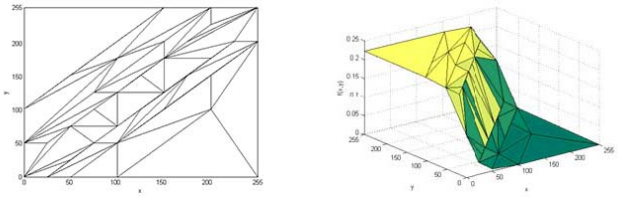

Fig. 9: Data-dependent tessellation of 36 selected points, avg. ΔE=0.0007.

3.1 Preliminary results of tessellation optimization

We also selected one plane of the printer data set (R=128) together with the corresponding *L*, a* and b** values. From the original 13×13=169 points we selected 25 points and performed a Delaunay and a data-dependent tessellation. If the two tessellations are used and applied to the $S-S_{sub}$ data set, the average and maximum approximation error achieved with the Delaunay tessellation is *avg.ΔL*=0.0015*, *maxΔL*=0.0022* versus *avg.ΔL*=0.0011*, *maxΔL*=0.0001* achieved with the data-dependent tessellation. The results achieved for a* and b* are *avg.Δa*=0.003* and *avg.Δb*=0.0049* using a Delaunay triangulation and *avg.Δa*=0.0025* and *avg.Δb*=0.004* using a data-dependent tessellation.

There are a couple of papers in the area of computational geometry which describe a data driven tessellation in 3D [6]. We are currently integrating our error metric into such a tesselator and testing whether the characterization data of a typical printer is suitable for data-dependent tessellation.

The final proposed algorithm is an iterative sequential algorithm that alternates point selection and tessellation. The first step selects a new point $p_j \in \{S-S_{sub}\}$ using the point selection algorithm form Section 2 and the tessellation from the previous iteration. The second step optimizes the tessellation using the data-dependent tessellation described in Section 3.

4. Conclusion

The proposed method to select a set of optimal sample points from a larger set enables a color transform to be specified in an efficient way, resulting in higher accuracy than achievable with the same number of uniformly placed sample points. This is beneficial to the processing time as well as to the resources (ink, media, and time) if those sample points are used for characterizations of a whole series of same model type printers. Initial tests suggest the newly developed error metric can also be used as a cost measure for a data-dependent tessellation of the selected sample points, at least for 2D data. Whether the data describing a complete printer characteristic is suited for a data-dependent tessellation will be reported in the final paper.

References

[1] I. Amidror: "Scattered Data Interpolation Methods for Electronic Imaging Systems: A survey", JEI **11**(*2*), 157-176 (April 2002).

[2] V. Monga, R. Bala, Sort-Damp-Select: "An Efficient Strategy for Look-Up Table (LUT) Design", Proc. IS&T/SID Color Imaging Conference, 2008.

[3] W.H. Press et. al.: Numerical Recipes, 3rd ed., Cambridge University Press, 2007, 1131-1142.

[4] N. Dyn, D. Levin and S. Rippa: "Data Dependent Triangulation for Piecewise Linear Interpolation", IMA J. Numer. Anal. **10**, 137-154 (1990).

[5] S. Rippa: "Long and Thin Triangles Can Be Good For Linear Interpolation", SIAM J. Numer. Anal. **29**(1), 257-270, (February 1992).

[6] J.R. Shewchuk: "Updating and Constructing Constrained Delaunay and Constrained Regular Triangulations by Flips", SoCG'03, June 8-10, 2003, San Diego, California, USA.

Author Biography

Ingeborg Tastl is a principal scientist at Hewlett-Packard Laboratories in Palo Alto, California. Her research interests are in the

areas of color reproduction and color management related to different printing technologies. She represents HP at the International Color Consortium (ICC). Before joining HP she worked at Sony's US Research Lab in San Jose, California. Ingeborg received her Masters degree in Computer Science from the Vienna University of Technology (1990) and a Ph.D. degree in Color Science from the same university (1995).

John Recker is a senior researcher in the Print Production Automation Lab at Hewlett-Packard Laboratories where he works on RIP technology for commercial presses. His research interests mainly revolve around software and hardware rendering and image processing systems for a variety of applications including printing, CAD and games. John received a B.S. Degree in Electrical Engineering from Cornell University (1982).

Yao Zhang is a PhD student in the Department of Electrical and Computer Engineering at the University of California, Davis. He received his B.S. in Electrical Engineering from the Beijing Institute of Technology. His research interests are in the area of GPU computing, especially in parallel algorithms for numerical linear algebra, and GPU architecture/ software optimization. He worked at HP-Labs as a summer intern in 2008.

Giordano Beretta is working at the Print Production Automation Lab at Hewlett-Packard Laboratories. He did his graduate work in Computational Geometry at ETH in Zurich, Switzerland, before joining Xerox PARC in 1984 to work on color reproduction. After a stint in strategic planning and becoming the technical advisor for Color at Canon, he joined HP. He is a member of ISCC and fellow of the IS&T and SPIE.